Seguridad Perimetral: Firewalls y NAT

Práctica 3

Objetivos

- Conocer las principales características de filtros de seguridad en redes de comunicaciones, basándose en Listas de Control de Acceso
- Comprender las ventajas que aporta la inspección de tráfico sobre las Listas de Control de Acceso estáticas
- Entender cómo afecta NAT tanto a la conservación de direcciones IP como a la protección de la red corporativa

3.1 Listas de Control de Acceso

Listas de Control de Acceso (Access Control List, ACL)

- Las ACLs son un elemento fundamental de las soluciones de seguridad en redes de comunicaciones
  - Configuración de firewalls → Proteger zonas de la red de accesos no autorizados
    - Los routers ISR de Cisco pueden actuar como firewalls al configurar ACLs

- **Una ACL es una lista secuencial de comandos `permit` o `deny`, que se aplican (o no) a los paquetes que atraviesan el router en base a información de las cabeceras de capa 3 y 4 del paquete**
  - Se pueden aplicar tanto a IPv4 como a IPv6 (en general a cualquier protocolo enrutado)

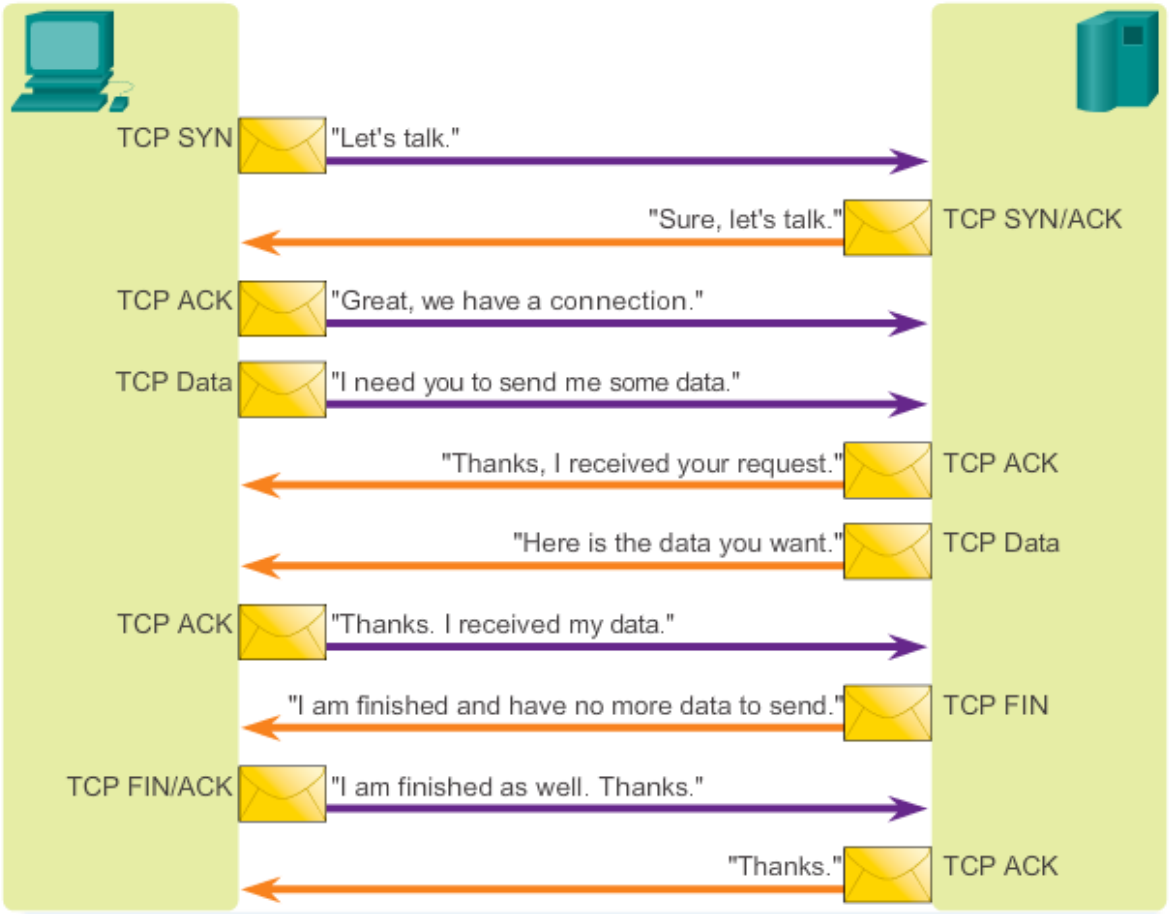
Funcionamiento de las ACLs

- Propósito de las ACLs:
  1. **Limitar el tráfico** para mejorar el rendimiento de la red
  2. Proporcionar **control de flujo** de tráfico para limitar el envío de actualizaciones de enrutamiento
  3. **Proporcionar un nivel básico de seguridad**
  4. **Filtrar tráfico** en base a múltiples criterios:
     - Clasificar el tráfico para ser procesado en VoIP
     - Filtrar el resultado de un “debug”
     - Definir el tráfico que va a ser traducido por NAT…
  5. **Controlar el acceso a servicios**

- Por defecto, el router no tiene ACLs configuradas → No filtra el tráfico
  - El tráfico que entra en el router se envía hacia el destino en base al contenido de la tabla de enrutamiento
- Al aplicar una ACL a una interfaz, el router debe evaluar si el paquete debe atravesar la interfaz a la que está asignada la ACL o no.

Funcionamiento de las ACLs (II)

- Repaso del establecimiento, mantenimiento y finalización de una conexión TCP

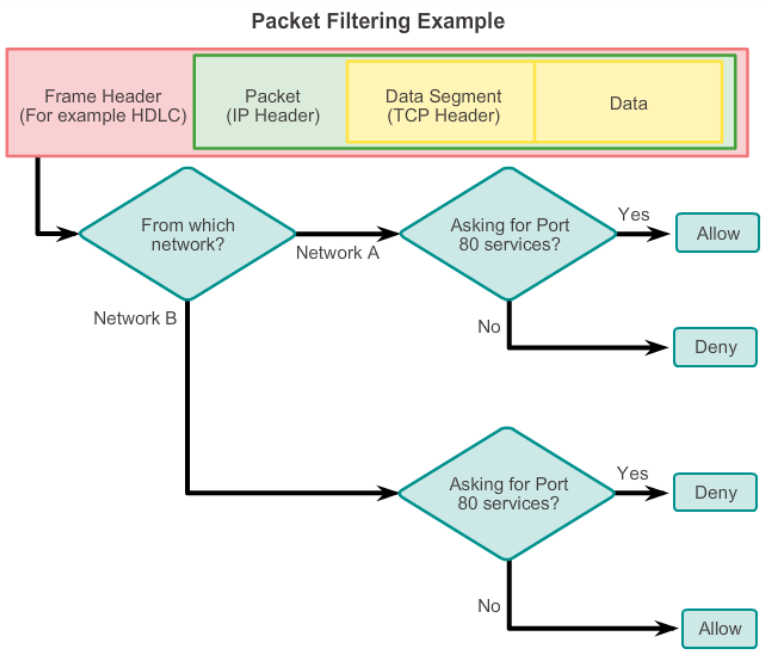


Funcionamiento de las ACLs (III)

- Funcionamiento del filtrado estático de paquetes:
  - El router controla el acceso entre redes analizando los paquetes entrantes y/o salientes, permitiéndolos o denegándolos, en base a determinados criterios, como:
    - **IP origen y destino**
    - **Campo protocolo**
    - **Datos específicos de los protocolos empaquetados en IP**
      - **Puertos TCP** origen y destino
      - **Puertos UDP** origen y destino
      - **Comandos ICMP**
  - El router analiza cada paquete individualmente, enviando (`permit`) o descartando (`deny`) cada uno en base a los criterios de coincidencia especificados en cada entrada de la ACL (Access Control Entry, ACE) o sentencia de la ACL
  - Cada una de las sentencias es evaluada secuencialmente. Cuando se produce una coincidencia con una de las sentencias, se ejecuta la acción asociada (`permit` o `deny`). Si no se cumple la condición, se examina la sentencia siguiente
  - Si no se produce coincidencia con ningún conjunto de condiciones → `Deny any`

Funcionamiento de las ACLs (IV)

- ACL:
  - Permite el acceso Web al tráfico de usuarios que provengan de la red A pero deniega el resto de servicios.
  - A los usuarios de la Red B, deniega el acceso Web y permite el resto de servicios
    - Si el paquete es un TCP SYN procedentede de A y dirigido al puerto TCP 80 → Permit
      - » El resto de tráfico de A → Deny
    - Si el paquete es un TCP SYN procedente de B y dirigido al puerto TCP 80 → Deny
      - » El resto de tráfico de B → Permit



Funcionamiento de las ACLs (V)

- Las ACLs actúan sobre los paquetes que "**entran en el router**" a través de la interfaz que tiene aplicada una ACL entrante y sobre los paquetes que "**salen del router**" a través de una interfaz que tiene una ACL saliente aplicada
- **Las ACLs no se aplican al tráfico generado por el propio router**
- Una vez creadas las ACLs se aplican a las interfaces de forma:
  - **Entrante**: Los paquetes que entran en el router a través de dicha interfaz son procesados por la ACL antes de ser enrutados
    - Suelen ser más eficientes
    - Se recomienda su uso
  - **Saliente**: Los paquetes que salen del router a través de dicha interfaz se procesan antes de ser transmitidos por la interfaz
    - Se recomiendan cuando hay que aplicar el mismo filtro a tráfico que proviene de múltiples interfaces
- La última sentencia de una ACL es siempre un **deny any** implícito → Si el paquete no coincide con ninguna entrada de la ACL, se bloqueará

Funcionamiento de las ACLs (VI)

- Existen dos tipos de ACLs en Cisco IOS para IPv4:
  - **Estándar**: Pueden filtrar únicamente por **dirección IP origen**

```
access-list 10 permit 192.168.30.0 0.0.0.255
```

  - **Extendidas**: Pueden filtrar por:
    - **IP origen y destino**
    - **Campo protocolo**
    - Datos específicos de los protocolos empaquetados en IP
      - **Puertos TCP origen y destino**
      - **Puertos UDP origen y destino**
      - **Comandos ICMP**

```
access-list 103 permit tcp 192.168.30.0 0.0.0.255 any eq 80
```

Funcionamiento de las ACLs (VII)

- Las ACLs estándar y extendidas se pueden crear tanto "**numeradas**" como "**nombradas**"

- ACLs numeradas:
  - ACLs estándar: 1 – 99; 1300 – 1999
  - ACLs extendidas: 100 - 199; 2000 – 2699

- ACLs nombradas:
  - El nombre puede contener letras y números
  - Se recomienda utilizar letras mayúsculas para el nombre de la ACL
  - El nombre no puede contener ni espacios ni signos de puntuación
  - Se pueden añadir y eliminar entradas de la ACL

Máscaras Wildcard

- Las sentencias de las ACLs utilizan máscaras "**wildcard**" para especificar qué bits de la dirección indicada deben ser comprobados y cuáles pueden ser ignorados
- La máscara "**wildcard**" es una cadena de 32 bits que permite definir qué bits de las direcciones origen o destino se van a comprobar:
  - Un **cero** en la máscara wildcard indica que **su bit correspondiente de la dirección va a ser comprobado para determinar si el paquete coincide** con las condiciones que se especifican en la sentencia
  - Un **uno** en la máscara wildcard indica que **su bit correspondiente de la dirección no va a ser comprobado para determinar si el paquete coincide** con las condiciones que se especifican en la sentencia

- **NO es obligatorio que la máscara wildcard sea la inversa de la máscara de subred**; Aunque en muchos casos, como cuando se van a permitir o denegar redes o subredes completas, si se da esta coincidencia.

Máscaras Wildcard (II)

- Ejemplos de máscaras Wildcard:
  - Comprobación de los 32 bits de una dirección IP

	Decimal	Binary
IP Address	192.168.1.1	11000000.10101000.00000001.00000001
Wildcard Mask	0.0.0.0.	00000000.00000000.00000000.00000000
Result	192.168.1.1	11000000.10101000.00000001.00000001

  - Comprobación de 0 bits de la dirección IP

Example 2

	Decimal	Binary
IP Address	192.168.1.1	11000000.10101000.00000001.00000001
Wildcard Mask	255.255.255.255	11111111.11111111.11111111.11111111
Result	0.0.0.0	00000000.00000000.00000000.00000000

  - Comprobación de los bits de red y subred

Example 3

	Decimal	Binary
IP Address	192.168.1.1	11000000.10101000.00000001.00000001
Wildcard Mask	0.0.0.255	00000000.00000000.00000000.11111111
Result	192.168.1.0	11000000.10101000.00000001.00000000

Máscaras Wildcard (III)

- Ejemplos de Especificación de Rangos de Direcciones:

Example 1

	Decimal	Binary
IP Address	192.168.16.0	11000000.10101000.00010000.00000000
Wildcard Mask	0.0.15.255	00000000.00000000.00001111.11111111
Result Range	192.168.16.0 to 192.168.31.255	11000000.10101000.00010000.00000000 to 11000000.10101000.00011111.11111111

Example 2

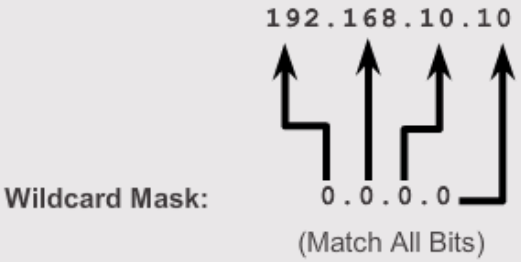
	Decimal	Binary
IP Address	192.168.1.0	11000000.10101000.00000001.00000000
Wildcard Mask	0.0.254.255	00000000.00000000.11111110.11111111
Result	192.168.1.0	11000000.10101000.00000001.00000000
	All odd numbered subnets in the 192.168.0.0 major network	

Máscaras Wildcard (IV)

- Palabras Clave:
  - **Host <IP>** → Comprobar todos los bits de la IP = <IP> 0.0.0.0
  - **Any** equivalente a 0.0.0.0 255.255.255.255

Example 1

- 192.168.10.10 0.0.0.0 matches all of the address bits
- Abbreviate this wildcard mask using the IP address preceded by the keyword **host** (**host 192.168.10.10**)



Example 2

- 0.0.0.0 255.255.255.255 ignores all address bits
- Abbreviate expression with the keyword **any**

Máscaras Wildcard (V)

Example 1:

```
R1(config)#access-list 1 permit 0.0.0.0 255.255.255.255
R1(config)#access-list 1 permit any
```

Example 2:

```
R1(config)#access-list 1 permit 192.168.10.10 0.0.0.0
R1(config)#access-list 1 permit host 192.168.10.10
```

Guías para la Creación de ACLs

- Ubicación de las ACLs:
  - En los routers que tengan que hacer función de firewall entre la red Interna y la red externa de la organización (Seguridad Perimetral básica)
  - En los routers que deban controlar el tráfico entre dos partes de la red interna. De este modo se puede controlar el tráfico que entra y sale de una parte de la red interna
  - Como conclusión, debe tenerse en cuenta que las ACLs deben colocarse en routers frontera, entre diferentes zonas de la red interna o entre las distintas zonas de seguridad de la red
  - Debe configurarse una ACL para cada protocolo de red (p.ej. IPv4 o IPv6) presente en los routers frontera
- Técnicamente es posible aplicar una ACL por:
  - Interfaz / Protocolo / Dirección

Guías para la Creación de ACLs (II)

- El uso de ACLs requiere sumo cuidado:
  - Los errores pueden generar cortes de servicio, necesidad de largos procesos de resolución de problemas y degradar el rendimiento de la red
  - Por ello, la creación de ACLs debe seguir un proceso cuidadoso:
    1. Basar las ACLs en la política de seguridad de la organización
    2. Preparar una descripción de lo que debe hacer la ACL
    3. Utilizar un editor de textos para crearla, editarlas y guardarlas
    4. Comprobar las ACLs en un laboratorio de pruebas antes de aplicarlas
    5. Tener un plan de recuperación

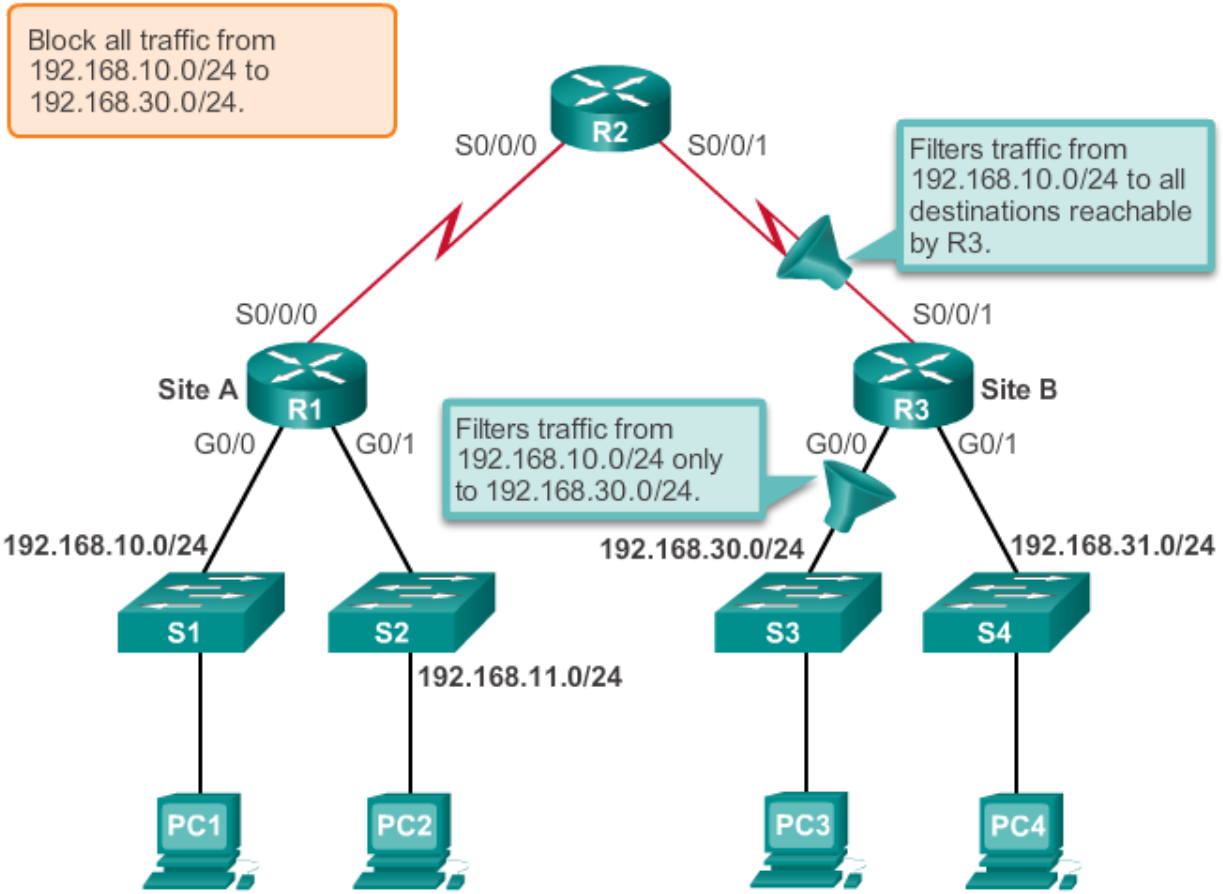
Guías de Ubicación de las ACL

- La ubicación de las ACLs puede afectar al rendimiento de la red
  - Intentar eliminar el tráfico no necesario lo más pronto posible
  - No limitar la conectividad entre redes que pueden intercambiar tráfico entre sí

- Reglas básicas (no son obligatorias, pero sí una recomendación a tener en cuenta):
  - ACL estándar: Deben colocarse próximas al destino
    - El único criterio de filtrado es la dirección IP de origen
      - Si se colocasen próximas al origen, dicho origen quedaría aislado
  - ACL extendidas: Deben colocarse lo más próximas al origen
    - Pueden definir flujos de tráfico de forma exacta
    - Pueden filtrar por IP origen y destino, protocolo, puerto, ….
      - Al colocarse cerca del origen, se evita que tráfico que, finalmente va a ser descartado, esté circulando por la red innecesariamente

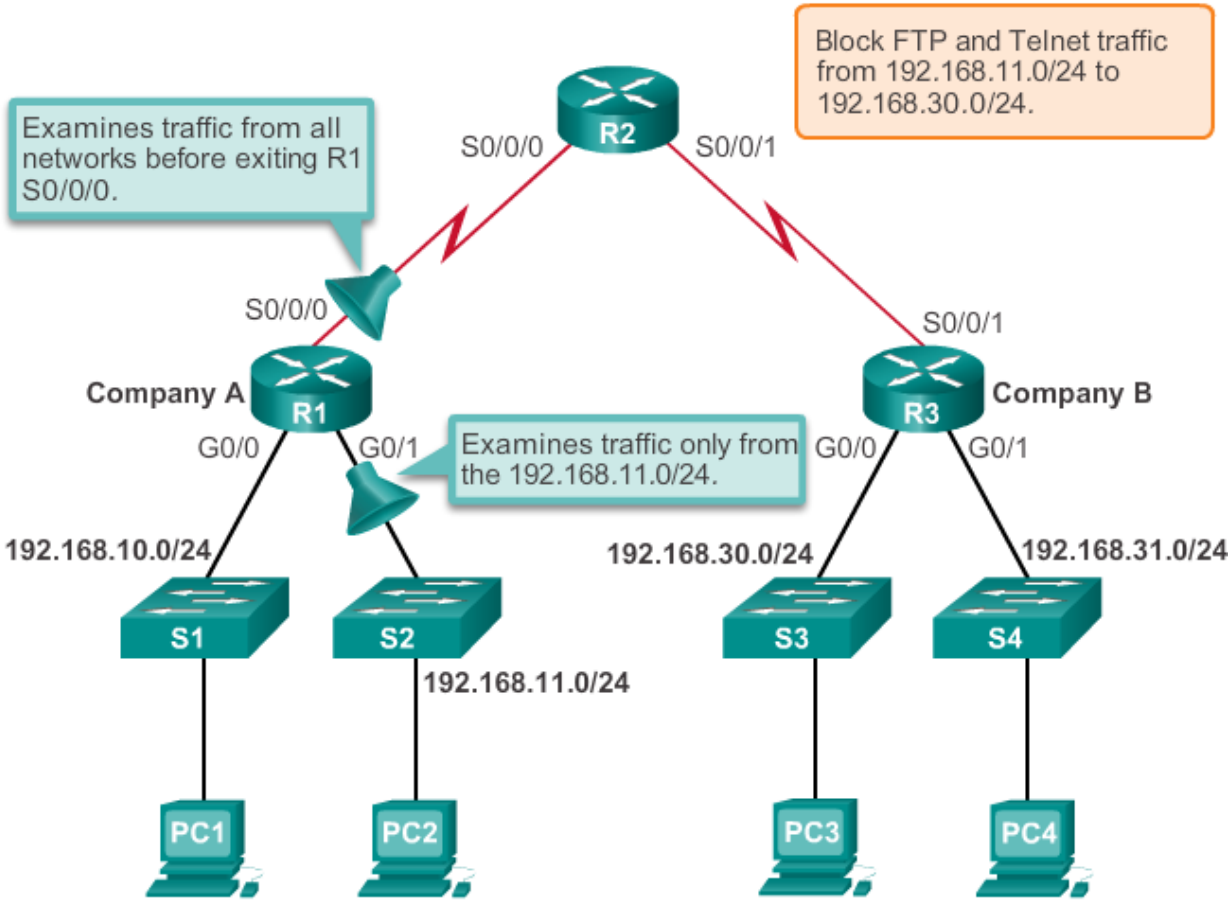
Guías de Ubicación de las ACL (II)

- Ejemplo de Aplicación de una ACL Estándar



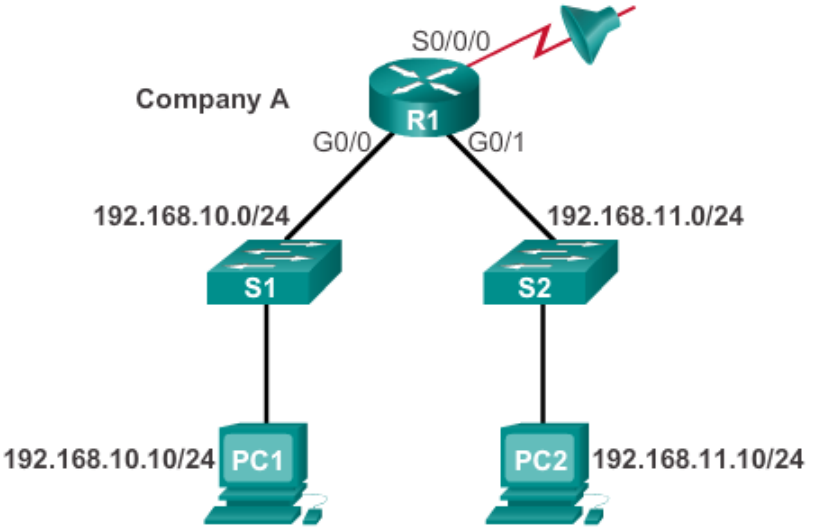
Guías de Ubicación de las ACL (III)

- Ejemplo de Aplicación de una ACL Extendida



Configuración de ACLs Estándar

- Cuando el tráfico entra (o sale) en un router que tiene aplicada una ACL entrante (o saliente) en una interfaz, las características de cada paquete se comparan con las entradas de la ACL (ACE) secuencialmente hasta que se produce una coincidencia con el criterio de selección. En ese momento, se ejecuta la orden asociada: Permitir o Denegar
- El resto de sentencias de la ACL no se comprueban
- Si no se produce coincidencia con ninguna entrada, el paquete se deniega
- Ejemplo:



ACL 1

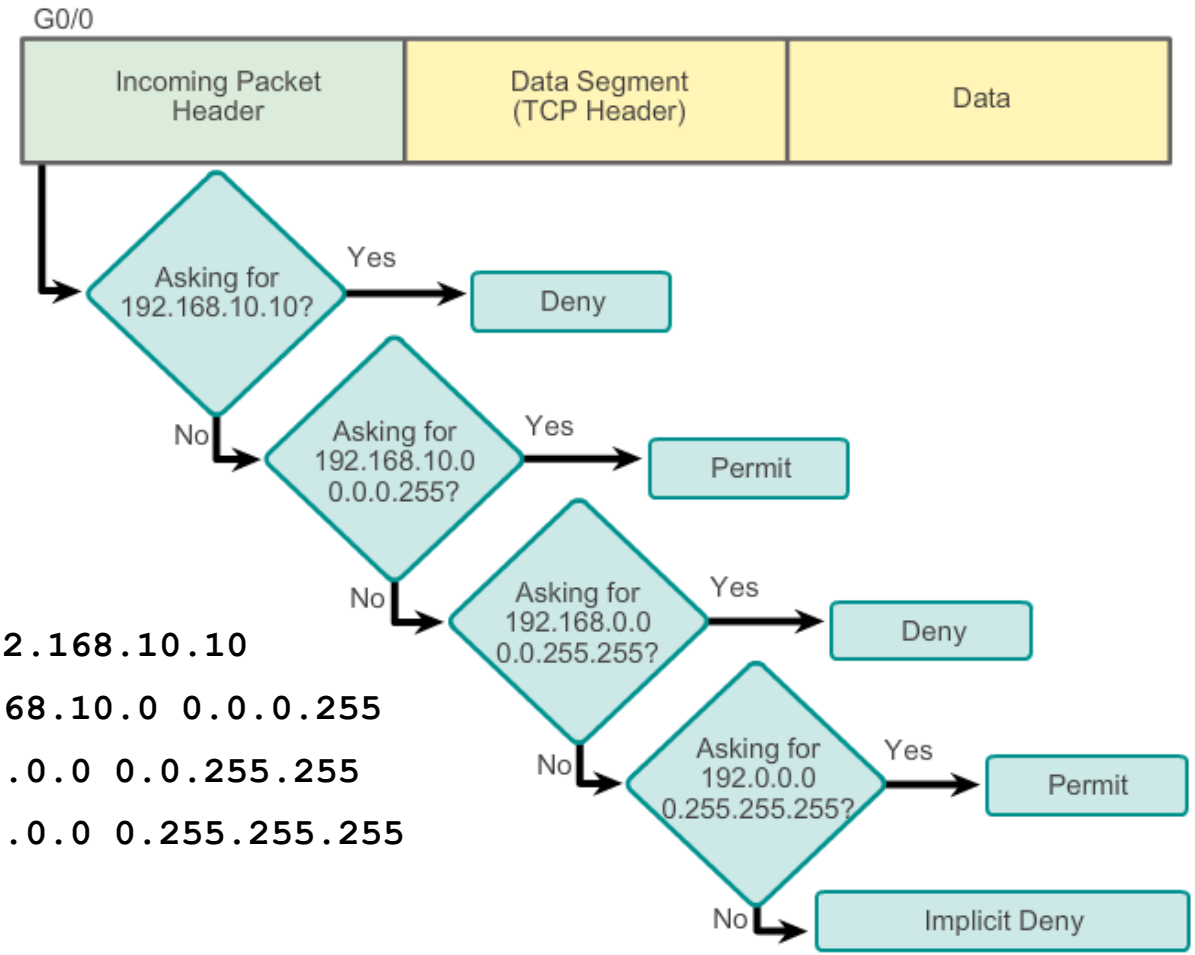
```
R1(config)#access-list 1 permit ip 192.168.10.0 0.0.0.255
```

ACL 2

```
R1(config)#access-list 2 permit ip 192.168.10.0 0.0.0.255
R1(config)#access-list 2 deny any
```

Configuración de una ACL Estándar (II)

- Cuando llega (o sale) un paquete al router a través de una interface que tiene una ACL aplicada, dicho paquete se procesa del siguiente modo



Ejemplo ACL

```
access-list 2 deny host 192.168.10.10
access-list 2 permit 192.168.10.0 0.0.0.255
access-list 2 deny 192.168.0.0 0.0.255.255
access-list 2 permit 192.0.0.0 0.255.255.255
```

Configuración de una ACL Estándar (III)

- Sintaxis:

```
Router (config)# access-list access-list-number deny permit
remark source [ source-wildcard ] [ log ]
```

- Para borrar una ACL, se utiliza el comando **no access-list** *access-list-number*
- Se puede usar el comando **remark** para documentar
- El orden de las sentencias es fundamental en una ACL

```
R1(config)#access-list 3 deny 192.168.10.0 0.0.0.255
R1(config)#access-list 3 permit host 192.168.10.10
```

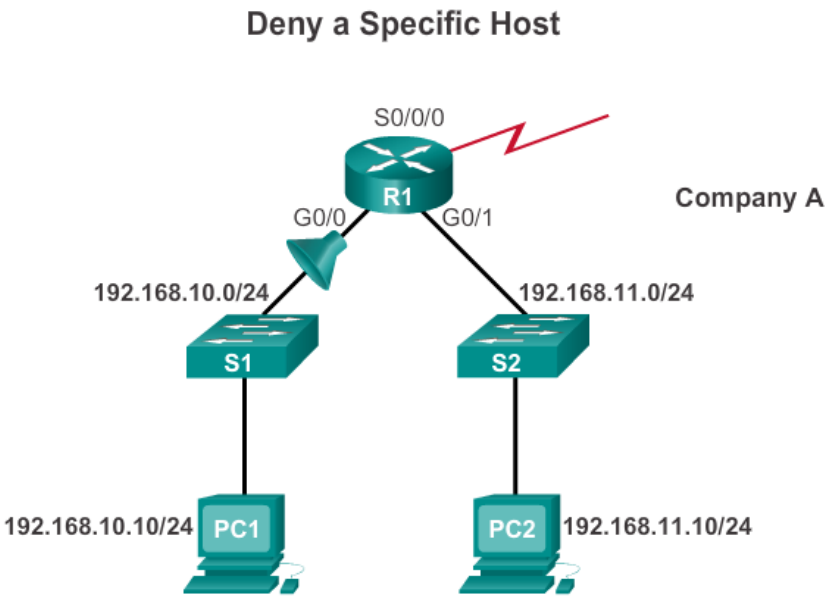
Aplicación de ACLs estándar a las Interfaces

- Para aplicar una ACL creada previamente a una interfaz se utiliza el comando: **`ip access-group`**
  - `Router(config-if)# ip access-group { access-list-number | access-list-name } { in | out }`

- Para eliminar la aplicación en la interfaz se debe utilizar el comando **`no ip access-group`**

- Al igual que el proceso de creación de la ACL es:
  - Crear la ACL
  - Aplicar ACL
- El proceso de eliminación debería ser:
  - Eliminar la aplicación de la ACL
  - Eliminar la ACL

Aplicación de ACLs estándar a las Interfaces (II)




```
R1(config)#no access-list 1
R1(config)#access-list 1 deny host 192.168.10.10
R1(config)#access-list 1 permit any
R1(config)#interface g0/0
R1(config-if)#ip access-group 1 in
```

Configuración de ACL estándar nombradas

1. **`Router(config)# ip access-list standard <nombre>`**

   `<nombre>`:
   - Case sensitive
   - Puede incluir letras y números, pero no signos de puntuación o espacios
   - Se recomienda la escritura en mayúsculas

2. **`Router(config-std-nacl)# permit | deny | remark <IP origen> <wildcard>`**

   `…`

   `exit`

3. **`Router (config)# interface <interface>`**

   `ip access-group <nombre> in | out`

4. Al igual que en las ACLs numeradas, pueden incluirse comentarios mediante la sentencia **`remark`**

Configuración de ACL estándar nombradas (II)

```
R1(config)#ip access-list standard NO_ACCESS
R1(config-std-nacl)#remark Do not allow access from Lab
workstation
R1(config-std-nacl)#deny host 192.168.11.10
R1(config-std-nacl)#remark Allow access from all other networks
R1(config-std-nacl)#permit any
R1(config-std-nacl)#interface G0/0
R1(config-if)#ip access-group NO_ACCESS out
R1(config-if)#
```

Modificación de las ACLs en IPv4: Editor de Textos Externo

- Cuando se configuran ACLs numeradas, sus comandos se añaden a la "`running-config`" pero no existe ninguna herramienta que permita editar dichas ACLs
- Las opciones disponibles son:

1. **Trabajar con un editor de textos externo (tipo notepad) y copiar y pegar la configuración obtenida de la edición**
   - Secuencia de Edición mediante editor externo

Configuration

```
R1(config)#access-list 1 deny host 192.168.10.99
R1(config)#access-list 1 permit 192.168.0.0 0.0.255.255
```

Step 1

```
R1#show running-config | include access-list 1
access-list 1 deny host 192.168.10.99
access-list 1 permit 192.168.0.0 0.0.255.255
```

Step 2

```
<Text editor>
access-list 1 deny host 192.168.10.10
access-list 1 permit 192.168.0.0 0.0.255.255
```

Step 3

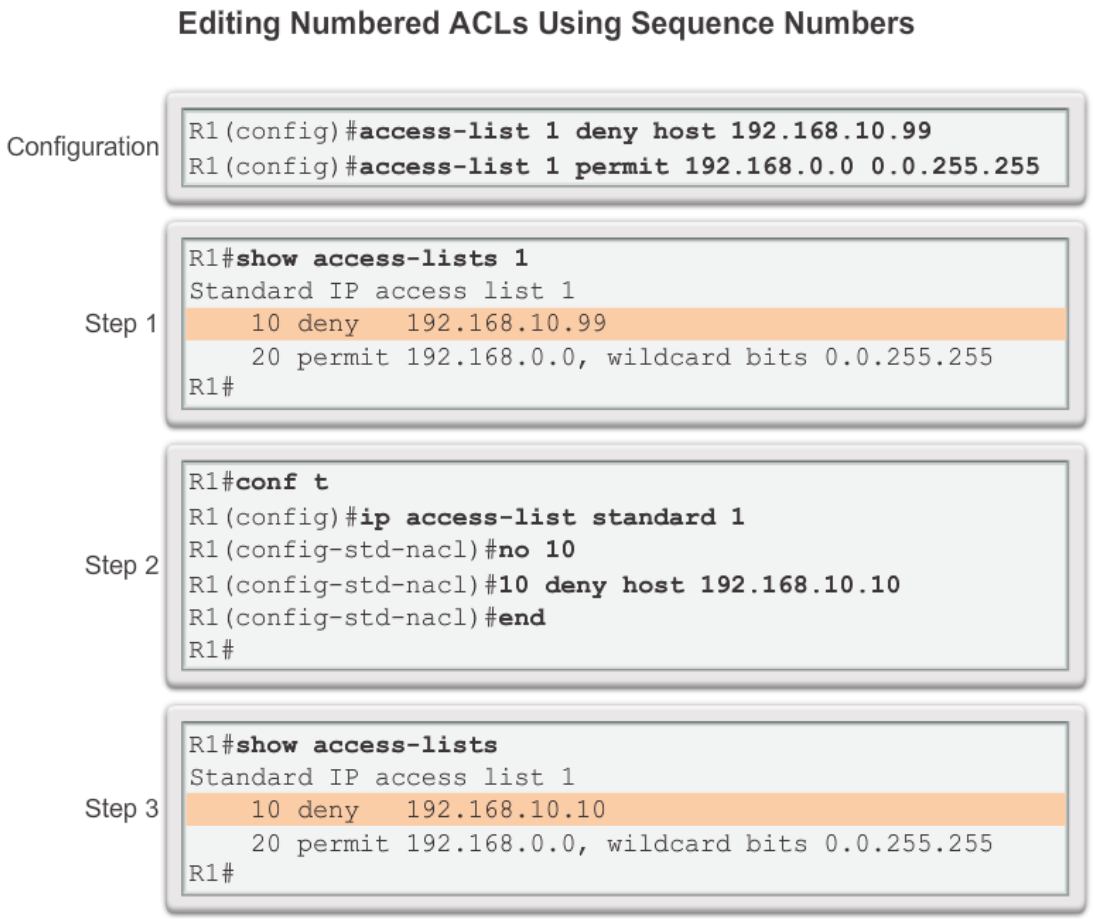
```
R1#config t
Enter configuration commands, one per line.  End with
CNTL/Z.
R1(config)#no access-list 1
R1(config)#access-list 1 deny host 192.168.10.10
R1(config)#access-list 1 permit 192.168.0.0 0.0.255.255
```

Step 4

```
R1#show running-config | include access-list 1
access-list 1 deny host 192.168.10.10
access-list 1 permit 192.168.0.0 0.0.255.255
```

Modificación de las ACLs en IPv4: Números de Secuencia

2. **Utilizar números de secuencia**



Modificación de las ACLs en IPv4: Inserción de una nueva ACE

```
R1#show access-lists
Standard IP access list NO_ACCESS
    10 deny   192.168.11.10
    20 permit 192.168.11.0, wildcard bits 0.0.0.255
R1#conf t
Enter configuration commands, one per line.  End with
CNTL/Z.
R1(config)#ip access-list standard NO_ACCESS
R1(config-std-nacl)#15 deny host 192.168.11.11
R1(config-std-nacl)#end
R1#show access-lists
Standard IP access list NO_ACCESS
    10 deny   192.168.11.10
    15 deny   192.168.11.11
    20 permit 192.168.11.0, wildcard bits 0.0.0.255
R1#
```

Note: The **no sequence-number** named-ACL command is used to delete individual statements.

Verificación de ACLs

```
R1# show ip interface s0/0/0
Serial0/0/0 is up, line protocol is up
  Internet address is 10.1.1.1/30
<output omitted>
  Outgoing access list is 1
  Inbound  access list is not set
<output omitted>

R1# show ip interface g0/0
GigabitEthernet0/0 is up, line protocol is up
  Internet address is 192.168.10.1/24
<output omitted>
  Outgoing access list is NO_ACCESS
  Inbound  access list is not set
<output omitted>
```

```
R1# show access-lists
Standard IP access list 1
    10 deny   192.168.10.10
    20 permit 192.168.0.0, wildcard bits 0.0.255.255
Standard IP access list NO_ACCESS
    15 deny   192.168.11.11
    10 deny   192.168.11.10
    20 permit 192.168.11.0, wildcard bits 0.0.0.255
R1#
```

Visualización de la Información Estadística

- Una vez asociada a una interfaz, comienza a visualizarse información estadística asociada a la ACL

```
R1#show access-lists
Standard IP access list 1
    10 deny   192.168.10.10 (4 match(es))
    20 permit 192.168.0.0, wildcard bits 0.0.255.255
Standard IP access list NO_ACCESS
    15 deny   192.168.11.11
    10 deny   192.168.11.10 (4 match(es))
    20 permit 192.168.11.0, wildcard bits 0.0.0.255
R1#
```

Output after pinging PC3 from PC1.

Matches have been incremented.

```
R1#show access-lists
Standard IP access list 1
    10 deny   192.168.10.10 (8 match(es))
    20 permit 192.168.0.0, wildcard bits 0.0.255.255
Standard IP access list NO_ACCESS
    15 deny   192.168.11.11
    10 deny   192.168.11.10 (4 match(es))
    20 permit 192.168.11.0, wildcard bits 0.0.0.255
R1#
```

ACLs Extendidas

- Para llevar a cabo un control del tráfico más preciso es necesario utilizar ACLs extendidas
  - Numeradas: 100 – 199; 2000 – 2699
  - Nombradas
- Comprueban:
  - IP de origen y destino
  - Campo protocolo
  - Otros campos de protocolos de capa superior:
    - Puertos en TCP y UDP
    - Tipos de comando ICMP

ACLs Extendidas (II)

- Permiten controlar el acceso a puertos y servicios específicos

Using Port Numbers

```
access-list 114 permit tcp 192.168.20.0 0.0.0.255 any eq 23
access-list 114 permit tcp 192.168.20.0 0.0.0.255 any eq 21
access-list 114 permit tcp 192.168.20.0 0.0.0.255 any eq 20
```

Using Keywords

```
access-list 114 permit tcp 192.168.20.0 0.0.0.255 any eq telnet
access-list 114 permit tcp 192.168.20.0 0.0.0.255 any eq ftp
access-list 114 permit tcp 192.168.20.0 0.0.0.255 any eq ftp-data
```

Configuración de ACLs Extendidas

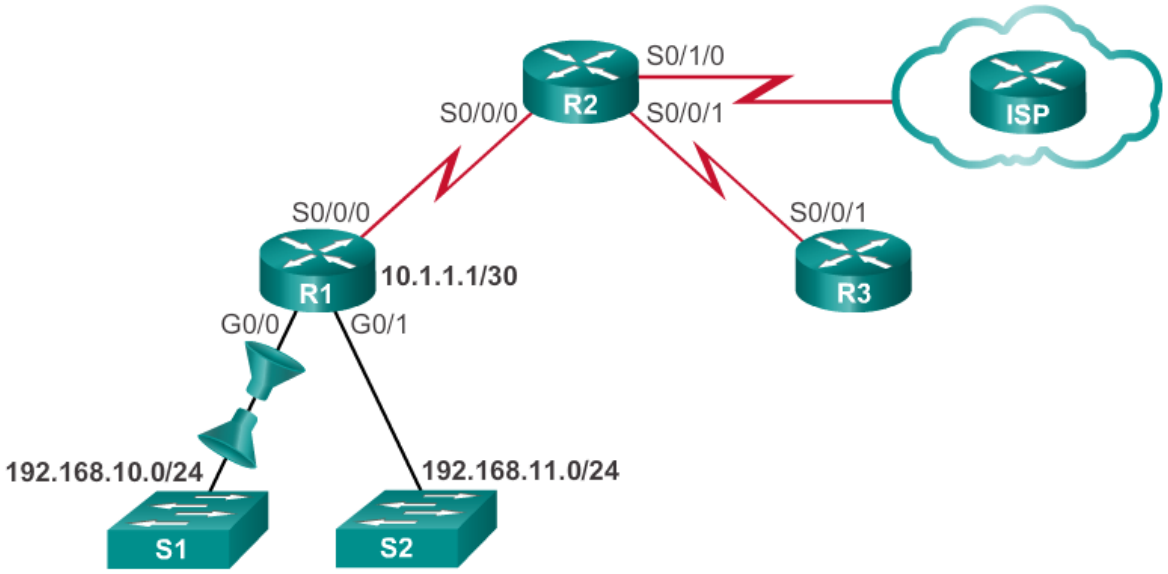
- El procedimiento de creación de ACLs extendidas es el mismo que en las ACLs estándar
- La sintaxis es muy diferente:

```
access-list access-list-number {deny | permit | remark}
protocol source [source-wildcard] [operator operand]
[port port-number or name] destination [destination-wildcard]
[operator operand] [port port-number or name][established]
```

  - La lógica interna de comprobación y ordenación de sentencias de las ACL estándar no existe en las ACLs extendidas
  - Las ACLs extendidas se aplican a las interfaces del mismo modo que las ACLs estándar

Configuración de ACLs Extendidas (II)

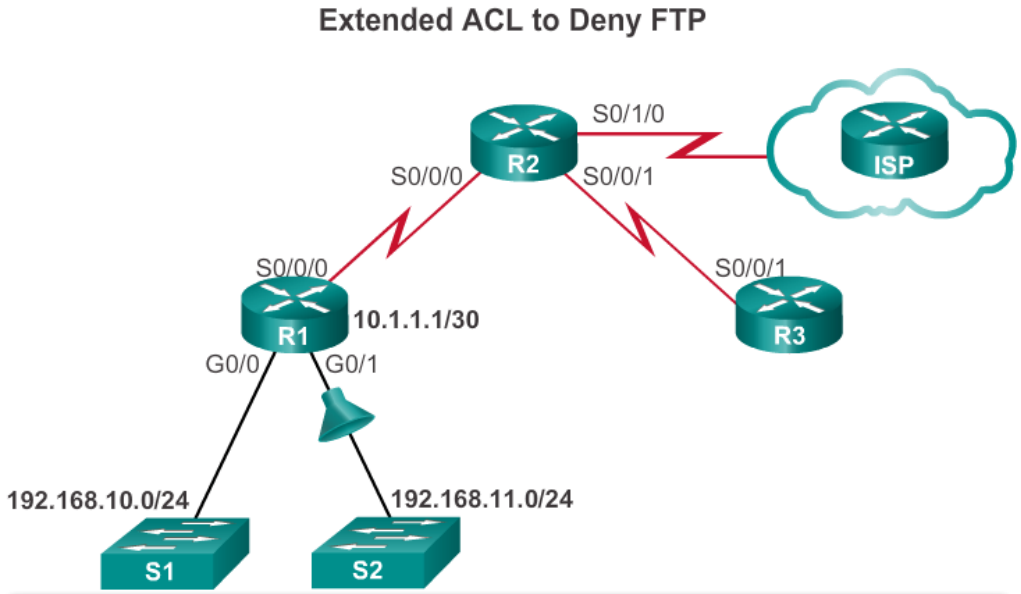
- Ejemplo I: Permitir a los usuarios de la red 192.168.10.0/24 navegar tanto con http como con https.
  - La ACL 103 se aplica en la int g0/0 en sentido entrante
  - El tráfico de entrada en la LAN 192.168.10.0 solamente se permite si es tráfico de conexiones TCP previamente establecidas




```
R1(config)#access-list 103 permit tcp 192.168.10.0 0.0.0.255 any eq 80
R1(config)#access-list 103 permit tcp 192.168.10.0 0.0.0.255 any eq 443
R1(config)#access-list 104 permit tcp any 192.168.10.0 0.0.0.255 established
R1(config)#interface g0/0
R1(config-if)#ip access-group 103 in
R1(config-if)#ip access-group 104 out
```

Configuración de ACLs Extendidas (III)

- Ejemplo II: En este ejemplo se deniega el tráfico FTP desde la subred 192.168.11.0 a la 192.168.10.0, pero se permite el resto del tráfico




```
R1(config)#access-list 101 deny tcp 192.168.11.0 0.0.0.255 192.168.10.0
0.0.0.255 eq ftp
R1(config)#access-list 101 deny tcp 192.168.11.0 0.0.0.255 192.168.10.0
0.0.0.255 eq ftp-data
R1(config)#access-list 101 permit ip any any
R1(config)#interface g0/1
R1(config-if)#ip access-group 101 in
```

Configuración de ACLs Extendidas (IV)

- Las ACLs extendidas nombradas se crean, fundamentalmente, igual que las ACLs estándar nombradas

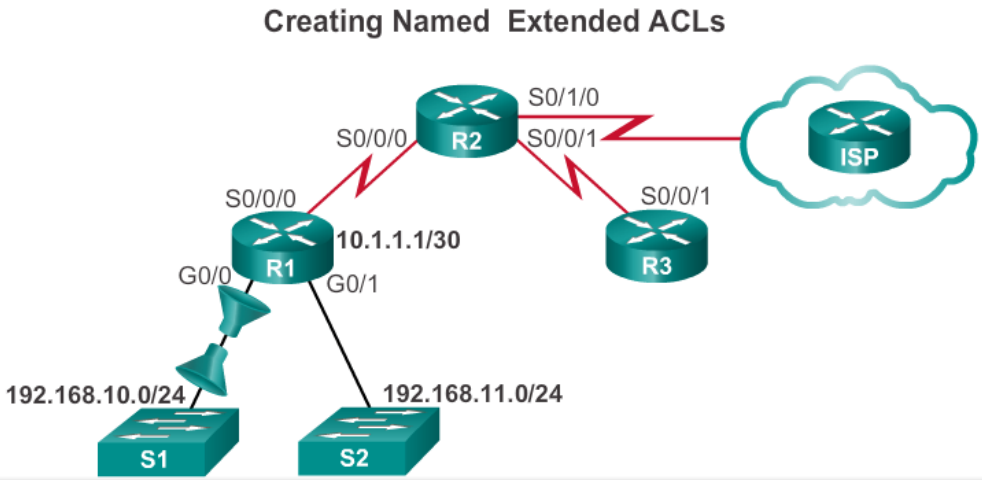
1. **`Router(config)# ip access-list extended <nombre>`**

2. Se introducen las sentencias **`permit, deny`** o **`remark`** necesarias

3. Una vez finalizada, se sale del modo de configuración de ACL extendida nombrada a modo privilegiado y se puede comprobar su estructura mediante **`show access-lists <nombre>`**

4. Para salvar la configuración **`wr`** o **`copy running-config startup-config`**

Configuración de ACLs Extendidas (V)




```
R1(config)#ip access-list extended SURFING
R1(config-ext-nacl)#permit tcp 192.168.10.0 0.0.0.255 any eq 80
R1(config-ext-nacl)#permit tcp 192.168.10.0 0.0.0.255 any eq 443
R1(config-ext-nacl)#exit
R1(config)#ip access-list extended BROWSING
R1(config-ext-nacl)#permit tcp any 192.168.10.0 0.0.0.255 established
R1(config-ext-nacl)#exit
R1(config)#interface g0/0
R1(config-if)#ip access-group SURFING in
R1(config-if)#ip access-group BROWSING out
```

Configuración de ACLs Extendidas (VI)

- Verificación de las ACL extendidas nombradas

```
R1#show access-lists
Extended IP access list BROWSING
    10 permit tcp any 192.168.10.0 0.0.0.255 established
Extended IP access list SURFING
    10 permit tcp 192.168.10.0 0.0.0.255 any eq www
    20 permit tcp 192.168.10.0 0.0.0.255 any eq 443
R1#
R1#show ip interface g0/0
GigabitEthernet0/0 is up, line protocol is up
  Internet address is 192.168.10.1/24
<output omitted for brevity>
  Outgoing access list is BROWSING
  Inbound  access list is SURFING
<output omitted for brevity>
```

Configuración de ACLs Extendidas (VII)

- Edición de las ACLs extendidas (mismo proceso que en las estándar):

  - **Método 1: Editor de Texto**
    - Eliminar la aplicación de la ACL a la interfaz
    - Copiar la ACL desde la "**`running-config`**" a un bloc de notas
    - Borrar la ACL mediante el comando **`no access-list`**
    - Modificar la ACL en el editor de textos
    - Copiar el resultado en el modo de configuración
    - Aplicar a la interfaz adecuada

  - **Método 2: Utilización de los números de secuencia**
    - Borrado de sentencias específicas
    - Inserción de ACEs en puntos específicos

Configuración de ACLs Extendidas (VIII)

- Inserción de ACEs en puntos específicos

```
R1# show access-lists
Extended IP access list BROWSING
    10 permit tcp any 192.168.10.0 0.0.0.255 established
Extended IP access list SURFING
    10 permit tcp 192.168.11.0 0.0.0.255 any eq www
    20 permit tcp 192.168.10.0 0.0.0.255 any eq 443
R1#
R1# configure terminal
R1(config)# ip access-list extended SURFING
R1(config-ext-nacl)# no 10
R1(config-ext-nacl)# 10 permit tcp 192.168.10.0 0.0.0.255 any eq
www
R1(config-ext-nacl)# end
R1#
R1# show access-lists
Extended IP access list BROWSING
    10 permit tcp any 192.168.10.0 0.0.0.255 established
Extended IP access list SURFING
    10 permit tcp 192.168.10.0 0.0.0.255 any eq www
    20 permit tcp 192.168.10.0 0.0.0.255 any eq 443
```

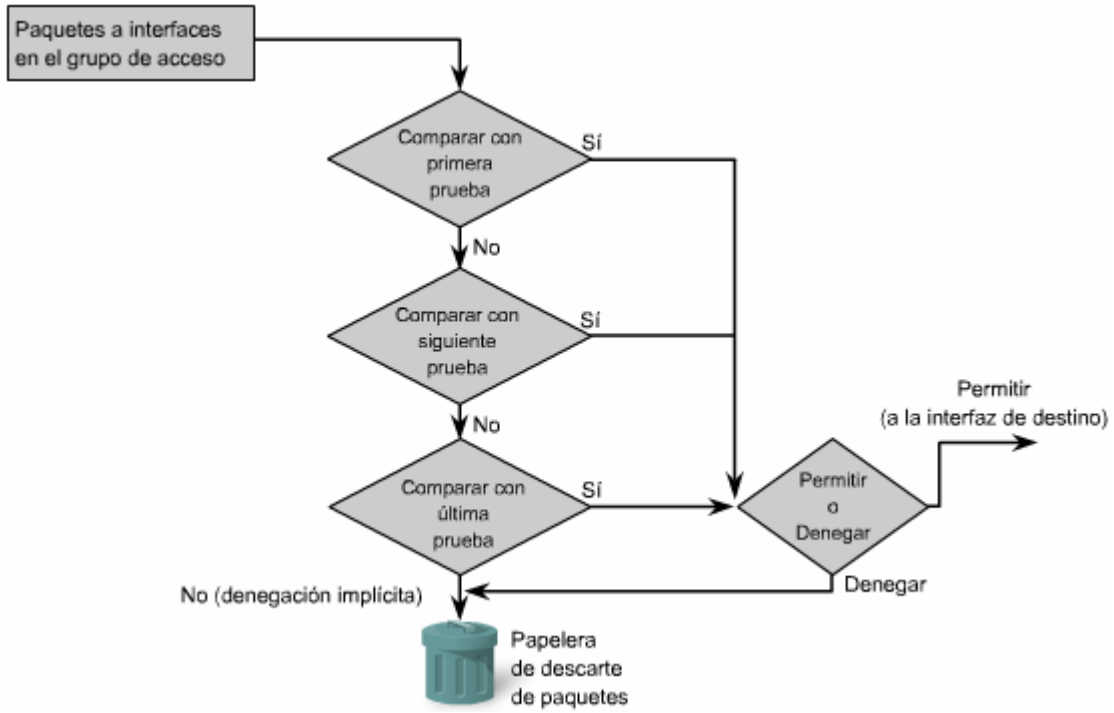
Should be 192.168.10.0

Filtrado de Paquetes: Listas de Control de Acceso

- Funcionamiento de una ACL entrante
  - Cuando un paquete llega al router a través de una interfaz con una ACL entrante asociada, la ACL se examina de arriba a abajo, línea a línea, buscando un patrón que coincida entre el criterio de filtrado establecido en cada línea con las características del paquete entrante.
    - Si se produce dicha coincidencia se aplica la acción asociada a la sentencia (permitir / denegar)
    - Si no se produce coincidencia con ninguna sentencia, el paquete se descarta (**`deny any`** / **`deny ip any any`** implícitos)
  - De manera predeterminada, un router no tiene ninguna ACL configurada y, por lo tanto, no filtra el tráfico. El tráfico que entra en el router es enrutado según la tabla de enrutamiento.

Filtrado de Paquetes: Listas de Control de Acceso

- Funcionamiento de una ACL de entrada en una interfaz

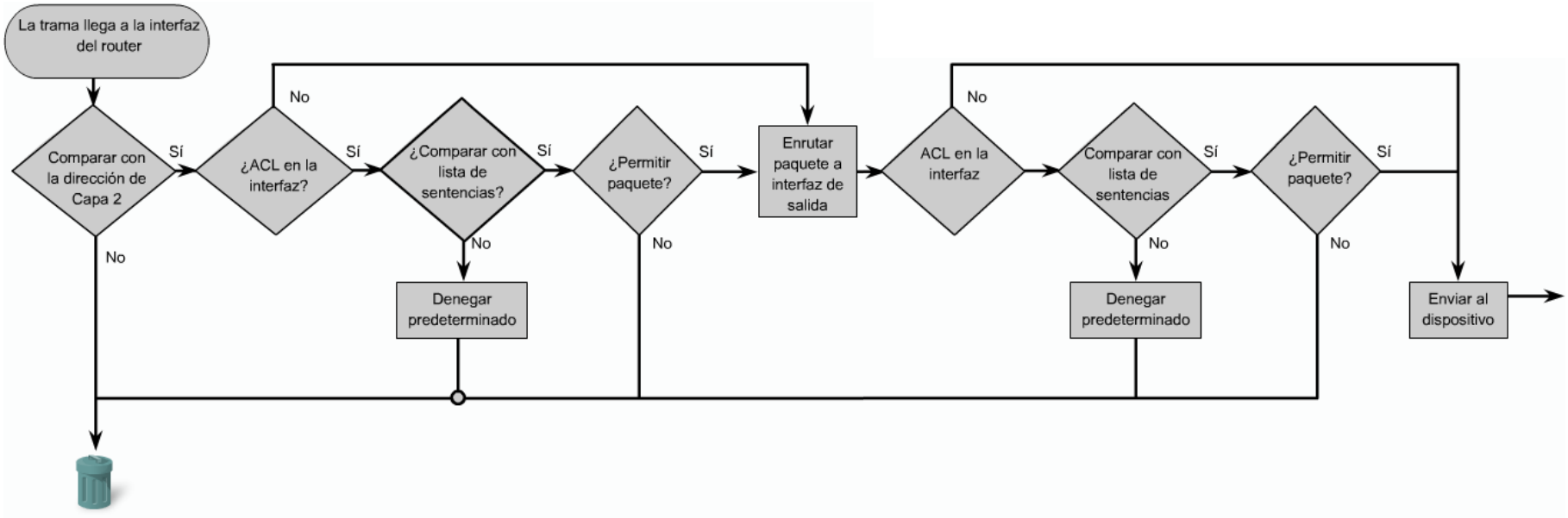


Filtrado de Paquetes: Listas de Control de Acceso (II)

- Funcionamiento de una ACL saliente
  - Cuando un paquete ha sido enrutado para ser enviado a través de una interfaz con una ACL saliente asociada, la ACL se examina de arriba a abajo, línea a línea, buscando un patrón que coincida entre el criterio de filtrado establecido en cada línea con las características del paquete entrante.
    - Si se produce dicha coincidencia se aplica la acción asociada a la sentencia (permitir / denegar)
    - Si no se produce coincidencia con ninguna sentencia, el paquete se descarta (**`deny any`** / **`deny ip any any`** implícitos)
  - De manera predeterminada, un router no tiene ninguna ACL configurada y, por lo tanto, no filtra el tráfico. El tráfico que entra en el router es enrutado según la tabla de enrutamiento.

Filtrado de Paquetes con Listas de Control de Acceso

- Ejecución de las ACLs en el proceso general de enrutamiento de un paquete IP en un router

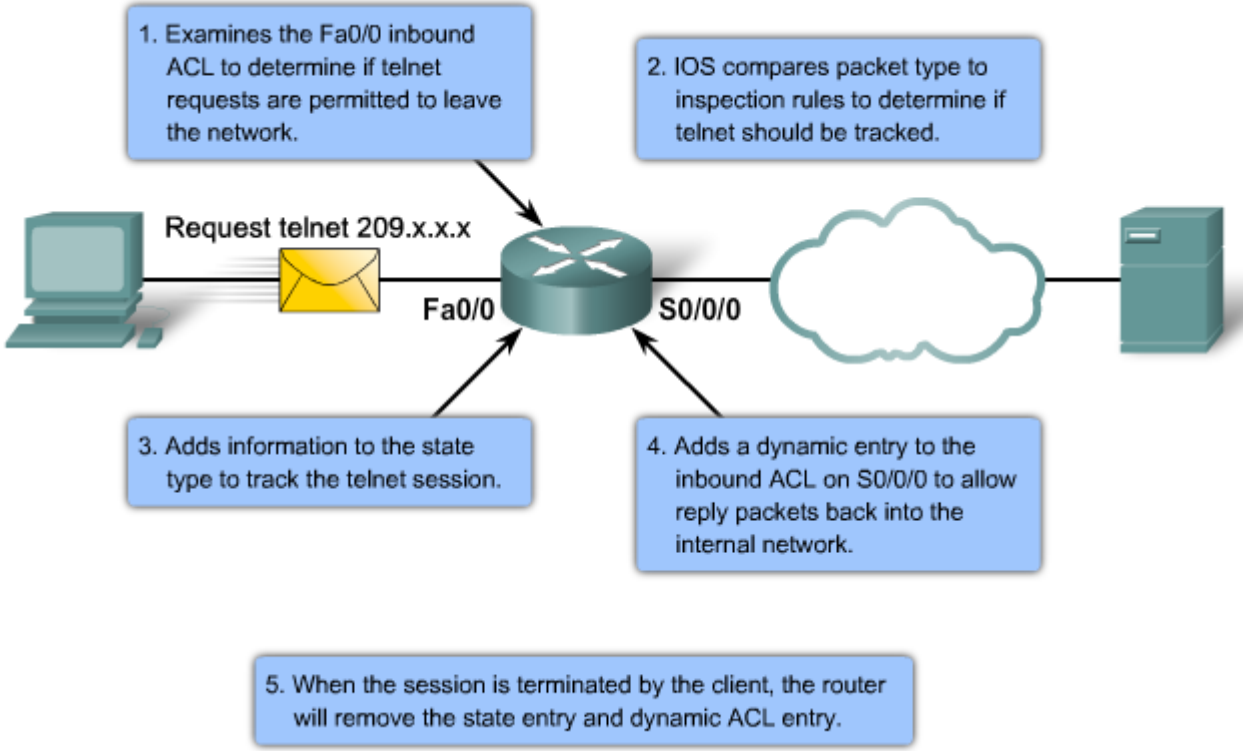


Control de Acceso Basado en el Contexto (CBAC)

Características CBAC

- CBAC es una solución de seguridad de firewall de capa de aplicación:
  - **Filtran de forma inteligente paquetes TCP y UDP basándose en información de las sesiones de los protocolos de capa de aplicación**
  - Proporciona filtrado con estado de capa de aplicación.
  - Diseñadas para filtrar tráfico de aplicaciones multimedia y protocolos que requieren de múltiples canales de comunicación tales como FTP, H.323, SIP, …
- Funcionalidades:
  - Filtrado de tráfico: Permitir tráfico TCP y UDP de retorno a través de un router cuando la conexión se inicia desde dentro de la red, mediante la creación de aperturas temporales en una ACL que de lo contrario negaría el tráfico
  - Inspección de tráfico: Analizan información de capa de transporte y aplicación para detectar ataques en estas capas
  - Detección de Intrusos
  - Generación de alertas y auditorías

Funcionamiento de las CBACs



- En el paso 3: la información de conexión se compara con las entradas en la tabla de estado. Si la conexión no existe actualmente, se añade la entrada. Si existe, el temporizador de inactividad para la conexión se restablece.
- En el paso 4: La apertura temporal sólo se activa durante el tiempo que la sesión está abierta. Estas entradas dinámicas ACL no se guardan en la memoria NVRAM.

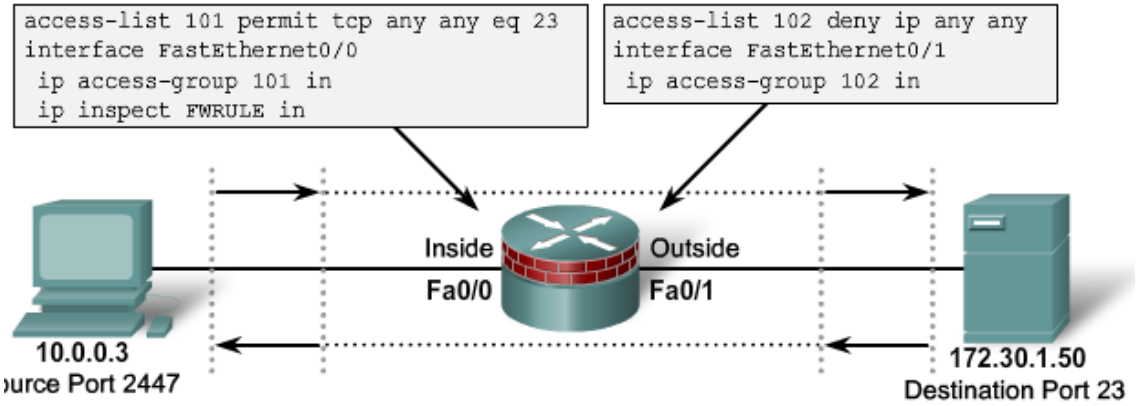
Funcionamiento de las CBACs (II)

- Las entradas temporales de las ACLs se crean como al inspeccionar el tráfico que sale de la red y se eliminan cuando termina la conexión o el tiempo de espera de inactividad de la conexión se ha alcanzado.
- Como en el caso de las ACL reflexivas, el administrador puede especificar qué protocolos se inspeccionan, así como el interfaz y en qué dirección se produce la inspección.
- Las CBACs son flexibles en la elección de la dirección en la que se inspecciona el tráfico.
  - En una configuración típica las CBAC se utilizan en el router perimetral o firewall para permitir el retorno de tráfico en la red.
  - Las CBACs también pueden ser configuradas para inspeccionar el tráfico en dos direcciones - dentro y fuera. Esto es útil cuando interesa la protección de redes en ambos lados puesto que hosts en ambas redes pueden iniciar ciertas conexiones y permitir que el tráfico vuelva a las fuentes correspondientes.

Funcionamiento de las CBACs (III)

- En CBAC la comprobación de tráfico saliente se especifica mediante reglas de inspección.
  - Una regla de inspección se aplica a un interfaz en una dirección.
  - Los paquetes que coinciden con la regla de inspección generan una entrada de ACL dinámica que permite al tráfico de respuesta retornar.
  - Si por el contrario un paquete es negado por la ACL de inspección el PKT se descarta
- Los valores "*timeout*" y "*threshold*" permiten controlar el tiempo que duran las sesiones tras un tiempo de espera o en el caso de que no hayan sido plenamente establecidas.
- Para controlar ataques DOS se pueden controlar los siguientes parámetros:
  - Número de sesiones semiabiertas TCP
  - Número de sesiones semiabiertas en un intervalo de tiempo
  - Número de sesiones semiabiertas por máquina

Funcionamiento de las CBACs (IV)



TCP traffic is inspected by FWRULE.

1. `ip inpsect FWRULE in`

Firewall creates a dynamic ACL allowing return traffic back through the firewall.

2. `access-list 102 permit tcp host 172.30.1.50 eq 23 host 10.0.0.3 eq 2447`

3. Firewall continues to inspect control traffic and dynamically creates and removes ACLs as required by the application. It also monitors and protects against application-specific attacks.

4. Firewall detects when an application terminates or times out and removes all dynamic ACLs for that session

Configuración de las CBACs

- Para crear una CBAC son necesarios los siguientes pasos:
  - Elegir una interfaz interna o externa.
  - Configurar una ACL IP en dicha interfaz.
  - Definir las normas o reglas de inspección.
  - Aplicar una regla de inspección a la interfaz.

Configuración de las CBACs (II)

- Elegir una interfaz interna o externa
  - La interfaz en la que se inician las sesiones se denomina interfaz interna. En consecuencia, el concepto de interna/externa se refiere a la dirección de la conversación.
  - Por ejemplo en un escenario con tres interfaces
    - Interfaz 1: red externa
    - Interfaz 2: DMZ
    - Interfaz 3: red interna
    - Puede interesar permitir sólo el acceso a servicios WEB y FTP en DMZ.
  - CBAC puede ser configurado en dos direcciones y en uno o varias interfaces.
  - Si buscamos proteger las redes de ambos lados, las CBAC se aplicarán en ambas direcciones.

Configuración de las CBACs (III)

- Configurar la ACL IP en la interfaz seleccionada
  - Las ACL se pueden configurar en un interfaz para tráfico entrante, saliente o ambos.
  - El administrador debe definir una ACL para controlar el tráfico de los protocolos que le interese inspeccionar. Asimismo deberá crear ACLs que explícitamente no permitan el tráfico que el administrador decida.
  - Si se desea inspeccionar un determinado tipo de tráfico dicho tráfico debe permitirse en todas las ACLs que se aplican al flujo de dicho tráfico.
  - Para implementar protecciones anti-spoofing es recomendable denegar cualquier tráfico entrante (en una interfaz externa) de una dirección de origen que coincida con una dirección de la red protegida.
  - También es una buena política prohibir mensajes de difusión con dirección de origen 255.255.255.255.

Configuración de las CBACs (IV)

- Definición de las reglas de inspección
  - El administrador debe definir las reglas de inspección para especificar los protocolos de capa de aplicación a inspeccionar en una interfaz. Normalmente, sólo es necesario definir una ACL con regla de inspección. Pero en ocasiones se establecen dos, una en cada dirección en un interfaz.
  - Una regla de inspección debe especificar el protocolo de capa de aplicación que se desea inspeccionar.
  - Las reglas de inspección incluyen opciones para el control de alertas y mensajes de auditoría.
  - La sintaxis del comando es:

```
Router(config)# ip inspect name inspection_name protocol [alert
{on | off}] [audit-trail {on | off}] [timeout seconds]
```

  - Ejemplo de una regla de inspección que inspecciona el tráfico de SMTP y FTP y que genera mensajes de alerta e información de auditoría.

```
ip inspect name FWRULE tcp timeout 300
ip inspect name FWRULE udp timeout 30
ip inspect name FWRULE icmp timeout 10
```

Configuración de las CBACs (V)

- Aplicación de una regla de inspección a un interfaz.
- La sintaxis es la siguiente:

  – `Router(config-if)# ip inspect inspection_name {in | out}`

- Hay dos principios fundamentales para la aplicación de normas de inspección y ACL en el router:
  - En la interfaz donde se inicia el tráfico, aplicar una ACL en dirección IN que permita sólo el tráfico buscado y aplicar la regla en dirección IN que inspeccione el tráfico buscado.
  - En otras interfaces, aplicar una ACL en dirección IN que deniegue todo el tráfico, excepto el tráfico que no ha sido inspeccionado por el firewall.

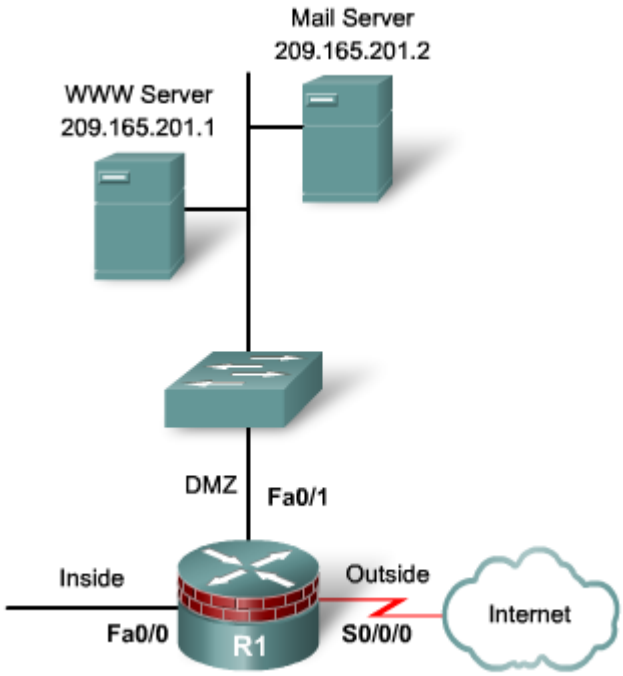
Configuración de las CBACs (VI)

- Ejemplo: Creamos una ACL que permite sesiones TCP, UDP, e ICMP y deniega el resto.

```
R1(config)# access-list 101 permit tcp 10.10.10.0 0.0.0.255 any
R1(config)# access-list 101 permit udp 10.10.10.0 0.0.0.255 any
R1(config)# access-list 101 permit icmp 10.10.10.0 0.0.0.255 any
R1(config)# access-list 101 deny ip any any
```

- La asociamos a la interfaz Inside fa0/0

```
R1(config)# interface Fa0/0
R1(config-if)# ip access-group 101 in
```



Configuración de las CBACs (VII)

- Continuación del Ejemplo:
  - Creamos una ACL extendida en la que se permite el tráfico SMTP y HTTP desde Outside a DMZ y el resto del tráfico se deniega.

```
R1(config)# access-list 102 permit tcp any 209.165.201.1 0.0.0.0 eq 80
R1(config)# access-list 102 permit tcp any 209.165.201.2 0.0.0.0 eq smtp
R1(config)# access-list 102 permit icmp any any echo-reply
R1(config)# access-list 102 permit icmp any any unreachable
R1(config)# access-list 102 deny ip any any
```

  - La asociamos al interfaz Outside en dirección IN

```
R1(config)# interface S0/0/0
R1(config-if)# ip access-group 102 in
```

Configuración de las CBACs (VIII)

- Continuación del Ejemplo:
  - Creamos las reglas de inspección para el tráfico TCP y UDP.

```
R1(config)# ip inspect name MYSITE tcp
R1(config)# ip inspect name MYSITE udp
```

  - Estas reglas de inspección se aplican al interfaz INSIDE en dirección IN.

```
R1(config)# interface Fa0/0
R1(config-if)# ip inspect MYSITE in
```

  - Las reglas de inspección crean automáticamente sentencias ACL temporales en la ACL aplicada a la interfaz externa para tráfico IN y conexiones TCP y UDP. De este modo se permite el tráfico TCP y UDP de respuesta a las peticiones generadas en el interior.
  - Para desactivar las CBACs del router podemos empelar el comando "**`no ip inspect`**" que elimina todas las CBAC, la tabla de estados y todas las entradas de ACL creadas por la CBAC. También resetea todos los *timeouts* y *threshold* a sus valores por defecto.
  - Al eliminar las CBACs los procesos de inspección ya no están disponibles y las ACLs se emplean sólo para filtrar.

Troubleshooting CBACs

- Existen comandos que permiten ver las entradas temporales creadas por una CBAC, la tabla de estadosy el fucnionamiento de al propia CBAC.
- Para ver la información acerca de inspecciones CBAC, utilice el comando

```
Router# show ip inspect [parameter]
```

Parameter	Explanation
name *inspection_name*	Limits the output of the display to only the inspection ruleset that you specified.
config	Displays the complete CBAC inspection configuration on the router.
interfaces	Displays the inspection rules activated on your router's interface(s).
sessions	Displays a summary of the connections in the CBAC state table.
sessions [detail]	Displays all the details for connections in the CBAC state table.
all	Displays all the information from the options listed in this table.

Troubleshooting CBACs (II)

- Ejemplos:
- El comando **`Router# show ip inspect sessions`** muestra la siguiente información

```
Established Sessions
Session 25A3378 (209.165.201.1:20)=>(192.168.1.2:32704) ftp-data
SIS_OPEN
Session 25A5AC2 (192.168.1.2:32703)=>(209.165.201.1:21) ftp SIS_OPEN
```

- El comando **`show ip access-list`** muestra las entradas dinámicas creadas por la ACL extendida entrante

```
Router# show ip access-list
Extended IP access list 100
    permit tcp host 209.165.201.1 eq 21 host 192.168.1.2 eq 32703 (24
    matches)
    permit tcp host 209.165.201.1 eq 20 host 192.168.1.2 eq 32704 (88
    matches)
```

Troubleshooting CBACs (III)

- Cuando buscamos información mucho mas detallada pordemos emplear el comando **`debug ip inspec`**

Parameter	Explanation
`tcp`	Displays TCP inspection events.
`udp`	Displays UDP inspection events.
`icmp`	Displays ICMP inspection events.
`application_name`	Displays inspection events for the specified application, such as TFTP or SMTP.
`events`	Displays CBAC events, including the processing of packets.
`object-creation`	Displays information about an entry being added to the state table.
`object-deletion`	Displays information about an entry being removed to the state table.
`function-trace`	Displays information about the software functions that CBAC calls.
`timers`	Displays information related to CBAC timers, such as information that the TCP or UDP idle timers are reached.
`detailed`	Displays information about all the CBAC processes on the router.

Configuración de NAT

Objetivos

- Comprender la necesidad del direccionamiento privado
- Discernir entre los diferentes tipos de configuraciones NAT: Estático vs. Dinámico
- Entender la necesidad de la existencia de NAT “sobrecargado” (NATP). Traducción de direcciones y puertos

Base Teórica

- Base teórica vista en clase

  – La traducción de direcciones de red (Network Address Translation) consiste en sustituir la dirección IP de origen (habitualmente), la dirección IP de destino (muy rara vez) o ambas cuando una paquete que cumple unas determinadas características es enviado a través de un dispositivo NAT como puede ser un router

  – Este mecanismo permite dispositivos de red configurados con IPs puedan acceder a redes públicas como Internet

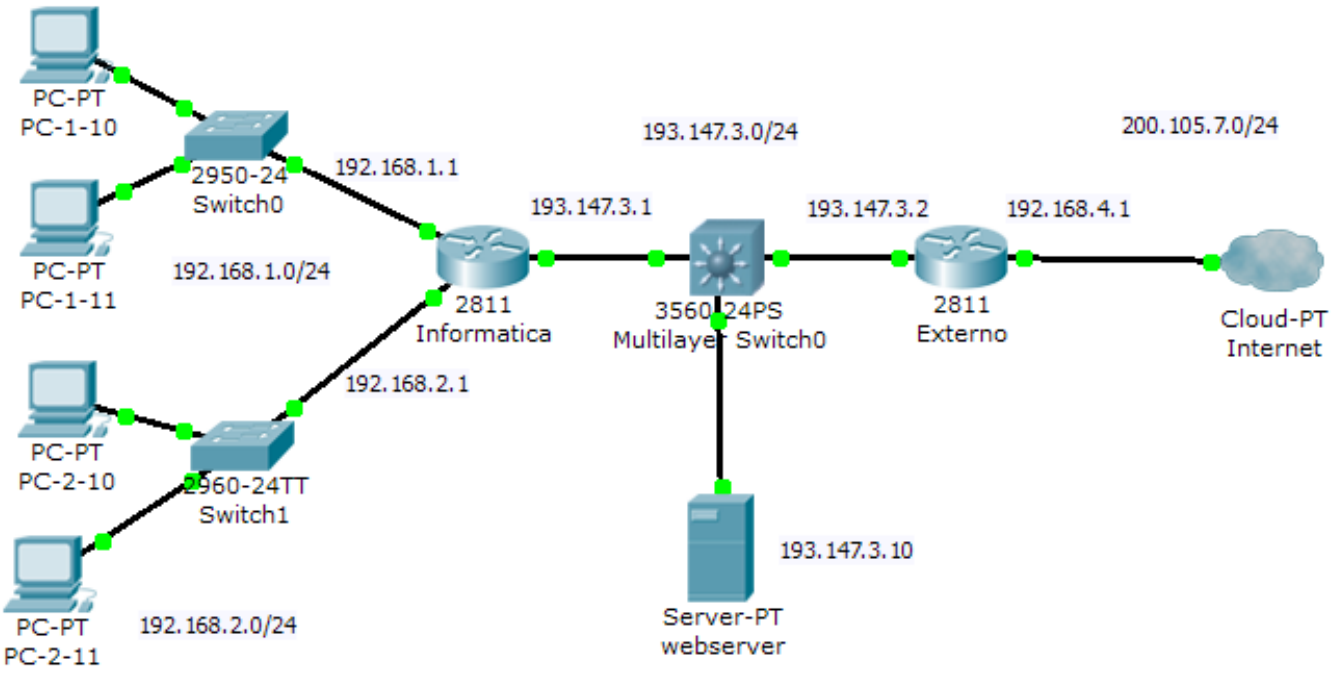
  – Rango de direcciones IP privadas (RFC 1918):
    - Clase A: 10.0.0.0 a 10.255.255.255 (1 red)
    - Clase B: 172.16.0.0 a 172.31.255.255 (16 redes)
    - Clase C: 192.168.0.0 192.168.255.255 (255 redes)

Base Teórica

- Tipos de NAT
  - NAT Estático: A cada dirección IP privada se le asigna una IP pública de forma administrativa
  - NAT Dinámico (sin sobrecarga): Existe un “pool” de direcciones privadas y un “pool” de direcciones públicas. Cuando un dispositivo con una IP privada necesita acceder a una red pública, solicita una dirección IP pública, si hay disponibles.
  - NAT Dinámico con Sobrecarga (Overloading) / Port Address Translation: En este caso una sola IP pública puede ser utilizada por múltiples IPs privadas, debido a que se traducen, además de las direcciones IP, los números de puerto de los protocolos de capa superior como TCP o UDP
  - Port Mapping / Port Forwarding: Es una asignación estática de IP + Puerto privados con una IP + Puerto público, lo que permite acceder desde la red pública (Internet) a determinados puertos de la red privada

Traducción de Direcciones: Configuración

- Ejemplo



Configuración de NAT estático

- Mapeo de direcciones IP
- Sintaxis:

```
Router(config)#ip nat inside source static <ip-local> <ip-global>
! Interfaz interna. Direccionamiento Privado
Router(config)#interface <tipo-int nº-int>
Router(config-if)#ip nat inside
! Interfaz externa. Direccionamiento Público
Router(config)#interface <tipo-int nº-int>
Router(config-if)#ip nat outside
```

Configuración de NAT (sin sobrecarga) estático:

- Mapeo de direcciones IP
- Ejemplo:

```
Router(config)#ip nat inside source static 192.168.1.10 193.147.3.200
! Interfaz interna.
! Direccionamiento Privado
Router(config)#interface f0/0
Router(config-if)#ip nat inside
! Interfaz externa.
! Direccionamiento Público
Router(config)#interface f1/0
Router(config-if)#ip nat outside
```

Configuración de NAT (sin sobrecarga) dinámico: Sintaxis

1. Definir el conjunto de direcciones públicas que van a ser utilizadas:

   ```
   Router(config)# ip nat pool <nombre> <ip-inicial> <ip-final> {netmask <máscara>|prefix-length <longitud-prefijo>}
   ```

2. Definir que direcciones IP privadas pueden ser traducidas, mediante una ACL

   ```
   Router(config)#access-list <n°> permit <origen> <wildcard-origen>
   ```

3. Relacionar el conjunto de direcciones IP privadas con el de direcciones IP públicas:

   ```
   Router(config)#ip nat inside source list <n° de ACL que define las Ips privadas> pool <nombre del pool de direcciones IP públicas>
   ```

4. Identificar la interfaz interna:

   ```
   Router(config)#interface <tipo-int n°-int>
   Router(config-if)#ip nat inside
   ```

5. Identificar la interfaz externa:

   ```
   Router(config)#interface <tipo-int n°-int>
   Router(config-if)#ip nat outside
   ```

Configuración de NAT (sin sobrecarga) dinámico: Ejemplo

1. Definir el conjunto de direcciones públicas que van a ser utilizadas:

```
Router(config)# ip nat pool Infor1 193.147.3.200 193.147.3.250 netmask 255.255.255.0
```

2. Definir que direcciones IP privadas pueden ser traducidas, mediante una ACL

```
Router(config)#access-list 1 permit 192.168.1.0 0.0.0.255
```

3. Relacionar el conjunto de direcciones IP privadas con el de direcciones IP públicas:

```
Router(config)#ip nat inside source list 1 pool Infor1
```

4. Identificar la interfaz interna:

```
Router(config)#interface f0/0
Router(config-if)#ip nat inside
```

5. Identificar la interfaz externa:

```
Router(config)#interface f0/1
Router(config-if)#ip nat outside
```

Configuración de NAT (**con sobrecarga**) dinámico o PAT

- Sintaxis. Opción 1: Sobrecargar un conjunto de direcciones IP públicas (grandes organizaciones)
  1. Definir el conjunto de direcciones públicas que van a ser utilizadas:

     ```
     Router(config)# ip nat pool <nombre> <ip-inicial> <ip-final> {netmask <máscara>|prefix-length <longitud-prefijo>}
     ```

  2. Definir que direcciones IP privadas pueden ser traducidas, mediante una ACL

     ```
     Router(config)#access-list <n°> permit <origen> <wildcard-origen>
     ```

  3. Relacionar el conjunto de direcciones IP privadas con el de direcciones IP públicas:

     ```
     Router(config)#ip nat inside source list <n° de ACL que define las Ips privadas> pool <nombre del pool de direcciones IP públicas> OVERLOAD
     ```

  4. Identificar la interfaz interna:

     ```
     Router(config)#interface <tipo-int n°-int>
     Router(config-if)#ip nat inside
     ```

  5. Identificar la interfaz externa:

     ```
     Router(config)#interface <tipo-int n°-int>
     Router(config-if)#ip nat outside
     ```

Configuración de NAT (**con sobrecarga**) dinámico o PAT

- Ejemplo
  1. Definir el conjunto de direcciones públicas que van a ser utilizadas:

     ```
     Router(config)# ip nat pool Infor1 193.147.3.200 193.147.3.250 netmask 255.255.255.0
     ```

  2. Definir que direcciones IP privadas pueden ser traducidas, mediante una ACL

     ```
     Router(config)#access-list 1 permit 192.168.1.0 0.0.0.255
     ```

  3. Relacionar el conjunto de direcciones IP privadas con el de direcciones IP públicas:

     ```
     Router(config)#ip nat inside source list 1 pool Infor1 OVERLOAD
     ```

  4. Identificar la interfaz interna:

     ```
     Router(config)#interface f0/0
     Router(config-if)#ip nat inside
     ```

  5. Identificar la interfaz externa:

     ```
     Router(config)#interface f0/1
     Router(config-if)#ip nat outside
     ```

Configuración de NAT (**con sobrecarga**) dinámico o PAT

- Sintaxis. Opción 2: Sobrecargar la dirección IP de la Interfaz Pública (PYME - Doméstica)
  1. Definir que direcciones IP privadas pueden ser traducidas, mediante una ACL

     ```
     Router(config)#access-list <n°> permit <origen> <wildcard-origen>
     ```

  2. Relacionar el conjunto de direcciones IP privadas con el de direcciones IP públicas:

     ```
     Router(config)#ip nat inside source list <n° de ACL que define las Ips privadas> interface <tipo número> OVERLOAD
     ```

  3. Identificar la interfaz interna:

     ```
     Router(config)#interface <tipo-int n°-int>
     Router(config-if)#ip nat inside
     ```

  4. Identificar la interfaz externa:

     ```
     Router(config)#interface <tipo-int n°-int>
     Router(config-if)#ip nat outside
     ```

Traducción de Direcciones: Configuración

- Configuración de NAT (**con sobrecarga**) dinámico o PAT: Ejemplo
  1. Definir que direcciones IP privadas pueden ser traducidas, mediante una ACL

     ```
     Router(config)#access-list 1 permit 192.168.1.0 0.0.0.255
     ```

  2. Relacionar el conjunto de direcciones IP privadas con el de direcciones IP públicas:

     ```
     Router(config)#ip nat inside source list 1 interface f1/0 overload
     ```

  3. Identificar la interfaz interna:

     ```
     Router(config)#interface f0/0
     Router(config-if)#ip nat inside
     ```

  4. Identificar la interfaz externa:

     ```
     Router(config)#interface f0/1
     Router(config-if)#ip nat outside
     ```

Configuración de "**Port-Mapping**":

- Sintaxis.

1. Relacionar la IP y puerto privado con la IP y puerto público:

```
Router(config)# ip nat inside source static [tcp|udp] <ip-local-interna>
<puerto-local-interno> <ip-global-interna> <puerto-global-interno>
```

2. Identificar la interfaz interna:

```
Router(config)#interface <tipo-int nº-int>
Router(config-if)#ip nat inside
```

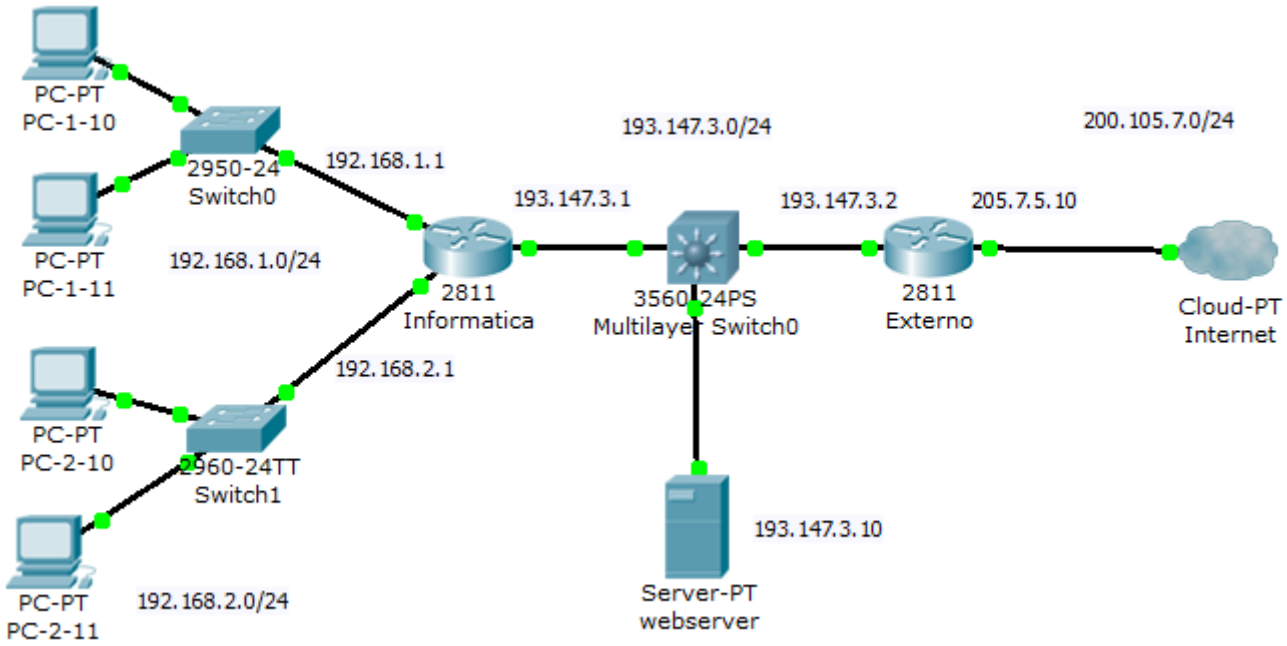
3. Identificar la interfaz externa:

```
Router(config)#interface <tipo-int nº-int>
Router(config-if)#ip nat outside
```

Configuración de "**Port-Mapping**":

- Ejemplo:

```
Router(config)# ip nat inside source static tcp 192.168.11.11 80 193.147.3.1
80
```



Desactivación de la Traducción de Direcciones

1. Eliminar el rol de interfaz interna y externa de aquellas en las que se haya configurado:

```
Router(config)# interface fastethernet 0/0
Router(config-if)# no ip nat inside
Router(config-if)# interface fastethernet 0/1
Router(config-if)# no ip nat outside
```

2. Limpiar la tabla de traducciones NAT

```
Router(config)# cliear ip nat translations *
```

3. Eliminar los comandos específicos de traducción y sus ACLs correspondientes:

```
Router(config)#no ip nat inside source list pool Infor1 overload
Router(config)#no ip nat pool Infor1 200.1.1.1 200.1.1.4 netmask
  255.255.255.248
Router(config)#no access-list 1
```

Comandos de Diagnóstico:

```
Router# show ip nat translations
Router# show ip nat statistics
Router# debug ip nat
```